EXPRESS MAIL NO. EV 331074531 US

PATENT APPLICATION
Attorney Docket No. D/A3175

# APPARATUS AND METHOD FOR GENERATING REUSABLE COMPOSITE COMPONENTS DURING DYNAMIC DOCUMENT CONSTRUCTION

## FIELD OF THE INVENTION

The present invention relates to the field of generating reusable document components (RDCs) and reusable underlays (RULs) in document rendering and, more particularly to methods for generating, caching, and using composite RDCs and RULs in systems which merge variable document data into static components.

## BACKGROUND OF THE INVENTION

Personalized print jobs contain both variable and reusable types of data. Variable data is typically printed on a form that remains static from page to page while the variable data itself is different on each form. Reusable data (reusable document components, or RDCs) appears multiple times within the same page, on different pages of the same job, or between different jobs.

A form, for instance, in a variable data job is considered to be a RDC. A form is also referred to as a reusable underlay (RUL) given the attribute that it matches the size of the page to be printed. The form is static because it is printed each time a completed form is printed. However, the information entered into the electronic form is variable content, which will typically differ for each person's completed form. Another example of a print job containing a mixture of variable and static content is a business letter with a standardized letterhead that is printed as part of the letter. The letterhead is static content across business letters, whereas the text of each letter is variable content. Similarly, in high print volume personalized advertising for mailing to customers or other large groups, the basic content is static while certain portions of the advertising are personalized based on information extracted from a customer database.

If cached, the cached RDCs will be used rather than re-rendering the component for the page. A given page can have more than one RDC placed on

it. Pages that have multiple RDCs currently require each RDC retrieved from the cache and merged to construct the page output. Merging the multiple RDCs along with the non-cached elements of the page is time consuming and lowers the page per minute throughput of the print path.

What is needed in the art is a system and method that renders and caches composite RDCs and composite RULs as they are interpreted for pages of a job.

## BRIEF SUMMARY

What is disclosed is a method for creating reusable composite components from interpreted pages of a rendered document during dynamic document construction. The method involves first obtaining a list of document components from the page and identifying any non-cached components. The individual reusable document components (RDCs) are cached having been rendered to their respective bounding box dimensions. These RDCs are then permuted into combinations of RDCs and each composite RDC is cached rendered relative to each other in a bounding box of sufficient size to adequately contain the combination. Then, combining RDCs to form composite RULs and caching the composite RULs rendered to full-page size. Any portion of each RDC of each composite RUL falling outside the boundary of RULs full-page size is clipped.

Also disclosed is a method for rendering pages having a combination of reusable components and non-cached components as the first step of assessing the page for the possibility of having an underlay-overlay pair. Then a cache of RULs is searched for underlays having the needed RDCs. If a matching RUL is not found in the cache then a new full page size RUL is created and cached. If a RUL is found or cached then a full-page size overlay is created from the non-cached components of the page. Lastly, the pages underlay (RUL) and overlay are used to render the page therefrom.

## BRIEF DESCRIPTION OF THE DRAWINGS

The preferred embodiments and other aspects of the invention will become apparent from the following detailed description of the invention when read in conjunction with the accompanying drawings which are provided for the

purpose of describing embodiments of the invention and not for limiting same, in which:

Figure 1 illustrates an apparatus embodiment of the invention for performing document construction;

Figure 2 illustrates an exemplary integration of the apparatus of Fig. 1 into a network-based printing station;

Figure 3 illustrates a flow chart of a method of how an RDC Repository works;

Figure 4 shows a fully constructed page with its RDCs and non-cached components from which the RDC composites are derived in accordance with the method of the present invention;

Figure 5 shows the RDC cache and the composite RDCs and RULs generated from the components of the page of Fig. 4;

Figure 6 illustrates how the cached composites of Fig. 5 are used to render subsequent pages; and

Figure 7 illustrates the rendering of subsequent pages in furtherance to Fig. 6.

## DETAILED DESCRIPTION OF THE SPECIFICATION

An appendix entitled: "PDSImager Design Specification to Support VIPP and DocuSP3.5", is attached hereto providing source code and other details in order to enable one skilled in this art to implement without undue experimentation the techniques of the present invention described herein.

With reference to Fig. 1, an exemplary apparatus 10 for performing document construction receives a document description 12 describing a print job, an electronic document, or the like. The document description 12 is preferably encoded in a page description language (PDL) representation such as a Variable data Intelligent Postscript Printware language (VIPP) or a Personalized Print Markup Language (PPML) based upon the Extensible Markup Language (XML) standard. The PDL specification includes at least one reuse hint that indicates a selected document component is a reusable document component (RDC).

A PDL such as VIPP can describe imaging resources potentially reused and advantaged if cached. The resources, in their cached form, are named "Reusable Document Components", or RDCs. A reusable component that is the same size as the page being rendered or displayed is called a "Reusable UnderLay" or RUL.

A VIPP page (a page from a VIPP job) can be comprised of cacheable RDCs and non-cached components. The cacheable RDCs can potentially be used on each or many of the following VIPP pages. The system renders and caches RDCs. When used on a page the RDCs can be retrieved from the cache and merged with the non-cached elements. A list of the RDCs and the non-cached elements for the page is built along with position information and used when merging the components for the final page output.

In general, document description 12 is processed by a PDL interpreter 14 that processes document portions or components. Typically, a document component is forwarded to a Compressor-Imager 16 that rasterizes the image into one or more pixel maps and compresses the pixel maps to reduce memory usage prior to printing. For color document components, the pixel maps are preferably continuous tone (contone) pixel maps. For black-and-white document components bit maps or half-tone pixel maps are preferred. Imager 16 constructs rasterized document pages from the document components. A buffer memory 18 accumulates rasterized document pages as they are constructed. The rasterized document pages are subsequently processed by a decompression module 20 that expands the compressed pixel maps. The resulting uncompressed rasterized document pages are forwarded to a print engine 22 that processes the pixel maps and controls a printing device 24 to produce document 26.

Instead of producing a printed document, the compressed pixel maps or document pages can alternatively or additionally be processed and used in other ways, such as being stored on a magnetic or optical disk for electronic viewing over a local network or over the Internet, transmitted via electronic mail, imported into another document or application, or the like.

The VIPP-2001 or other PDLs in which the document description 12 is encoded support reusable document hints that enable reuse of document components, which are rendered multiple times within a single document. These hints can be applied by the generating application to promote reuse of document components both within a selected document and also across documents.

When the PDL interpreter 14 encounters a reusable document hint in the document description 12, the PDL interpreter 14 references an internal state, such as a RDC index 30, to determine whether or not the document component corresponding to the hint has previously been rasterized and stored in Repository 32 which includes RDCs accumulated during past processing of other documents as well as RDCs generated as the present print job 12 is processed.

If the PDL interpreter 12 locates an RDC identification in the RDC index 30 that corresponds to the document component with the hint, the PDL interpreter communicates with the RDC repository to: (1) verify that the RDC is still contained in the RDC repository; and (2) command the RDC repository to preserve the RDC until it is accessed. The PDL interpreter also sends the RDC identification to Imager 16 rather than sending the actual document component. Imager 16 receives the RDC identification and communicates with the RDC repository to retrieve the corresponding compressed pixel map which is then stored in memory 18 and further processed substantially similarly to the processing of ordinary (i.e., not reusable) document components.

If, however, the PDL interpreter does not locate the document component with the hint in the RDC index, it communicates both the document component and an RDC identification to the Imager which processes the document component to generate a compressed pixel map to be stored in memory. The Imager additionally sends the compressed pixel map along with the RDC identification to the RDC repository 32 for possible reuse. The RDC is stored in the RDC repository as a compressed pixel map (e.g., compressed contone data for typical color document components). Along with the compressed pixel map, the RDC repository preferably stores selected additional information pertaining to

the RDC such as a compression mode, an RDC size, the RDC identification, and a lifetime parameter.

The lifetime parameter indicates how long the RDC repository should store the RDC. The lifetime is determined by the PDL interpreter 14 and communicated along with the RDC identification to the Imager 16, which then forwards the lifetime to the RDC repository 32. The lifetime can correspond to a lifetime indicated along with the reuse hint in the document description 12. Alternatively, the lifetime is selected by the PDL interpreter 14 based on the type of RDC, the nature of the document description 12, parameters of the corresponding print job, or similar information. The lifetime may be set to the termination of the present print job described by the document description or alternatively set to 'permanent' indicating that the RDC should not be deleted except by an express command of a user.

The RDCs of the RDC repository are stored in a long-term non-volatile storage 34 and/or a short-term RDC memory cache 36 associated with the repository. Preferably, a user can manage the RDC repository including storage 34, 36 via GUI 38. Optionally, the RDC repository also performs RAM cache cleanup when the cache is close to capacity to remove those RDCs that have not been accessed recently. The cleanup frees up space in the cache for RDCs which are currently being frequently accessed while relegating less-frequently accessed RDCs to longer term storage 34. Optionally, the RDC repository references the lifetime parameter associated with a RDC to perform automated deletions of RDCs from the RAM 36, and also from storage 34.

With continuing reference to Fig. 1 and with further reference to Fig. 2, apparatus 10 is integrated into printing station 50 connected with network 52 that includes computers 54. Printing station 50 receives print jobs such as the exemplary document description 12 via network 52.

The memories 34, 36 are suitably embodied as allocated portions of general-purpose memory 56 and RAM 58, respectively, of printing station 50. Memories 56, 58 also store information such as printing parameters 60 for the printing station and buffer memory 18 (Fig. 2 as memories 18V, 18R). Memories

34, 36 are allocated to the RDC repository or is optionally user adjustable through GUI 38 embodied on PC 66.

Other elements of apparatus 10 in Fig. 1, such as the PDL interpreter 14, the Imager 16, decompression module 20, and print engine 22, are also integrated into the printing station; not shown in Fig. 2.

## HOW AN RDC REPOSITORY WORKS

With reference to Fig. 3, a method 80 for employing the RDC repository 32 during document construction is described. PDL document 82 is examined in parsing step 84 to identify document components for rasterization. Each document component is checked 86 to determine whether the document component has a reuse hint associated therewith. If there is no reuse hint then that component is rasterized 88 and compressed 90 to produce component 92. If, however, decision 86 finds that a reuse hint. the PDL interpreter checks RDC index 30 in step 100 to determine whether the RDC has been previously encountered and whether a corresponding pixel map is presently stored in the RDC repository. If the RDC has been encountered previously, the PDL interpreter pings the RDC repository in step 102 in order to verify that the pixel map is still in the RDC repository. At decision 104 the PDL interpreter decides how to proceed based upon a response of the RDC repository. If the RDC repository locates a corresponding pixel map it marks it as read-only and communicates a confirmation to the PDL interpreter that the pixel map is available. In this case, retrieval 106 is performed to retrieve the corresponding compressed pixel map 108 from the RDC repository. If the retrieval was from storage 34, the RDC (which is now the most recently used RDC) is mapped into cache in step 110. The retrieved compressed pixel map 108 is sent to downstream components 94 in substantially the same manner as if the Imager 16 had just generated it. If, however, decision step 100 finds that the RDC is not indexed in the RDC index 30, or if the decision step 104 finds that the pixel map corresponding to the RDC is no longer in the RDC repository then steps 88 and 90 are performed. The resulting compressed pixel map 92 is communicated to the downstream components 94.

Additionally, however, if at decision 112 it is recognized that the newly created pixel map corresponds to a reusable document component then in 114 the created pixel map 92 is communicated to the RDC repository to be stored in storage 34. Since this newly created RDC is also the most recently used RDC, it is preferably mapped into the cache 36. If the RDC has a short lifetime, e.g., limited to the present print job, the RDC is optionally placed in the cache only and not stored.

COMPOSITE RDCs and RULs

A composite RDC is an RDC made up of multiple RDCs in order to provide improved performance whereby the final page is a merging of fewer components, the non-cached components and the composite. A composite RDC is reused if the same component combination and same relative positions is found on a subsequent page.

What follows is an example of how composite RDCs and RULs are created given some made up page PDL scenario involving multiple reusable components (RDCs).

Attention is now directed to Fig. 4. A composite RDC is created from the RDCs in the component list of page PDL 116 as it is interpreted to have the following RDCs and non-cached components: RDC1 at 118 with BBox1 at position X1, Y1; RDC2 at 120 with BBox2 at X2, Y2; RDC3 at 122 with BBox3 at X3, Y3; and non-cached components (NCC) at 124.

With reference being made to Fig. 5, the resulting RDC cache 126 derived from page PDL 116 of Fig. 4 at the time the RDC combination is first rendered are shown. Stored in cache 126 are RDCs 118, 120, and 122 of Fig. 4 rendered to their respective bounding box (BBox) dimensions. Single RDCs are cached in a size defined by the VIPP.

Also stored in cache are the composite RDCs, shown collectively at 128, rendered to their relative positions. Cached combination RDC1, RDC2, at 130, is stored rendered relative to each other into a BBox having a size, which can adequately contain this combination. Cached combination RDC2, RDC3, at 132, is stored rendered relative to each other into a BBox having a size, which

contains this combination. Cached combination RDC1, RDC3, at 134, is stored relative to each other into a BBox containing this pair. Lastly, the combination RDC1, RDC2, and RDC3, at 136, is stored rendered relative to each other into a BBox that can contain these three.

A combination of RDCs is also preferably combined in their relative positions to form a composite RUL, shown collectively at 138 of Fig. 5. Caching RDCs as composite RULs allows subsequent pages that use the RDCs in the same placement to render non-cached components of the page as an overlay. A composite RUL is a RUL made up of multiple RDCs and/or multiple RULs. A composite RULs is a variant of a composite RDC in that the composite RUL is created to provide a faster page rate given the Imager's ability to create overlay output from the non-cached elements of a page and the marker/print path's ability to merge underlays and overlays on the fly. The composite RULs rendered to full-page size are shown at 140 through 152. Note that RDC2, at 120 of Fig. 4, is clipped in composite RULs 142, 146, 148, and 152 because a composite RUL is specifically created to have the size of the page being rendered. Thus, any part of any RDC falling outside the page boundaries is subsequently clipped.

Attention is now directed to Fig. 6, which illustrates how the cached composites of Fig. 5 are used to render subsequent pages. Page 154 requires cached RDC Composite 136 and non-cached components 124 of page 116 of Fig. 4. Subsequently rendered page 156 requires cached RDC Composite 130 and non-cached components 124. Page 158 requires RDC component 118 and NCC 124. However, as many VIPP print jobs are created having the same combinations of RDCs repeated from page to page, the RDC composite that combines all of the RDCs into a composite RUL (rather than the lesser perturbations of RDCs) provide a performance boost because the composite RULs allow these repeated page types to become underlay-overlay pairs. As such, rendered page 158 is itself assessed for the possibility of generating an underlay-overlay pair. Since underlay 140 is already cached, in 138 of Fig. 5,

the non-cached component is turned into overlay 162 and cached. Page 160 utilizes cached overlay 152 and overlay 163.

With reference now being made to Fig. 7, if a following page is assessed to have an RDC combination that has a cached composite RUL for the needed RDCs, the non-cached components for the page are also preferably turned into an overlay. This allows a performance boost in that non-cached components alone tend not to take long to render and no combining is needed in the rendering stage. As with the RDCs, the combining is moved to a later display or print stage and, if the data format of the underlay and overlay allow easy merging and both components are the same size, combining can be done at a high rate.

Thus, page 164 of Fig. 7 merges cached RDC Composite 132 with the non-cached component (NCC). Page 166 uses underlay 150 with non-cached objects as an overlay 168. Page 170 uses underlay 152 with non-cached objects as an overlay 172. Page 174 merges cached RDC composite 134 with the non-cached component (NCC).